Logical foundations of knowledge-based recommender systems: A unifying spectrum of alternatives

Logical foundations of knowledge-based recommender systems: a unifying spectrum of alternatives

Federica Cena, Luca Console, Fabiana Vernero

Dipartimento di Informatica, Università di Torino
Corso Svizzera 185, I-10149 Torino (Italy)

Abstract

The aim of this paper is to provide logical foundations for knowledge-based recommender systems, for which, unlike other problem solving tasks, a comprehensive formalization is not yet available. This goal is justified by the need to compare recommenders based on the way they use knowledge to generate recommendations and, consequently, on the underlying semantics of the recommendation process itself. Moreover, since the here adopted logical formalization has been borrowed from other tasks such as diagnosis, many interesting results and opportunities can be transposed from such tasks to recommendation.
While we do not aim at proposing a new recommendation generation technique, we believe that our formalization will be the basis for unifying different approaches to knowledge-based recommendation, revealing their semantics and offering a conceptual framework to compare them. In fact, the framework covers different variations of knowledge-based recommendation, such as context-aware, constraints-based, package and group recommendation, as well as recommendation based on negative preferences.

Keywords: Recommender Systems, Logical Foundations, Formal framework covering various approaches.

1. Introduction

Recommendation is a prominent area of research within artificial intelligence: by aiming at predicting items which users may like or deem useful, recommender

systems serve the double goal of supporting users' decision making process and helping businesses increase their revenues, by implementing strategies such as cross-selling [2] and long tail marketing [9]. Given these premises, it is not surprising that recommender systems have been applied to domains as diverse as books, movies, travel accommodations, restaurants, jobs and people to follow in social networks, with a great variety of approaches proposed in the last decades. With so many options to choose from, being able to compare and assess different recommenders is a crucial task for system designers. Traditionally, recommendation algorithms have often been evaluated based either on prediction-based metrics such as the MAE, MSE or RMSE, whose aim is to estimate how good a recommender is at guessing user ratings on proposed items, or on information retrieval-related metrics, such as precision, recall or accuracy, which measure the system ability to distinguish between relevant and non-relevant items [46]. Embracing a more user-centric perspective, alternative metrics such as novelty, serendipity or diversity have been proposed to capture the system ability to actually satisfy user needs [46].

In contrast, the possibility of comparing recommenders based on *the way they use knowledge to generate recommendations* has attracted less attention in the literature. Such an approach, however, would provide system designers with the opportunity of choosing among different recommendation generation strategies or alternative recommendations based on their underlying semantics and, therefore, on their suitability for the specific recommendation problem they are trying to solve.

Unlike other popular problem-solving tasks such as diagnosis, configuration or planning, limited efforts have been devoted to providing a comprehensive formalization and knowledge-level analysis of recommendation up to now. Analysing and formalizing a problem solving task at knowledge level (following Newell's terminology [36]), i.e., abstracting from details and aspects concerning other levels, can provide very interesting insights on the task itself and allows the creation of frameworks for analysing the conceptual differences between alternative approaches.

Let's take the case of diagnosis. Research on logical foundations for this task started in the late '80s and resulted in a number of *theories of diagnosis* (see the collection of papers in [27]). Logical characterizations provided formal semantics to the notion of solution to a diagnostic problem. They allowed to analyse and compare different approaches, not necessarily implemented in logical terms, at conceptual and knowledge level, abstracting from details and aspects concerning other levels. Several properties of diagnostic problem solving could be studied, including aspects concerning the correctness and completeness of problem solvers, their complexity as well as methods for the efficient implementation of problem solvers. Thus, logical characterizations started multiple threads of research in different directions to cope with a number of dimensions beyond the purely logical bases (such as time, uncertainty, dynamicity ...).

The goal of this paper is to plant the seeds to start the same threads in the area of recommendation, providing a comprehensive formalization of this task. To this aim, we focus on knowledge-based recommender systems, a specific line of research where the recommendation results from problem solving on an explicit knowledge base which relates user features and preferences to categories and/or items to be recommended. Systems which do not rely on a knowledge base, such as collaborative filtering recommenders, are left out of the scope of our framework not because we believe they are less effective or valuable, but merely for the sake of simplicity, in that they resort to different types of semantics.

In particular, in this paper we show that the same framework we proposed in the past to provide a unified logical theory of diagnosis can be adopted also in the case of knowledge-based recommendation. Thus, we do not aim at introducing yet another approach to recommendation based on logic, but rather we aim at using logic to provide unifying foundations for knowledge based recommendation, covering a variety of approaches in the literature. In particular, we will extend and adapt to recommendation a number of interesting results and opportunities:

- First of all, we provide a formalization for the concept of "knowledge-based

recommendation problem", considering its main ingredients (Section 3).

- We then define the notion of solution to a recommendation problem, showing that it cannot be defined uniquely and singling out a spectrum of definitions (Section 4).

- We show that such a spectrum of definitions can be the basis for comparing alternative approaches at the knowledge level and discuss criteria to select among them, relating different definitions to the form of recommendation which is required and to the characteristics of the available knowledge base and data. We also discuss how to deal with incomplete and imprecise knowledge bases (Section 4.1).

- We characterize recommendation as a process which progressively refines a set of candidate solutions, suggesting, at each step, the best data to be gathered in order to discriminate among them (Section 5). According to this view, we define criteria for ranking alternative solutions.

- We then show how various forms of recommendation (negative preferences-based, context-aware, constraints-based, package and group recommendation) can be characterized as special cases in our framework (Section 6).

- Finally, in the conclusion we suggest that the problem solvers developed in the diagnostic community can be adopted also for knowledge-based recommendation, thus supporting a number of steps beyond the logical foundations discussed in the paper (Section 6).

2. Background

Recommender systems emerged in the Nineties as information search and filtering tools aimed at supporting users in finding items they might like, be interested in or otherwise deem useful [6].

2.1. Prediction techniques

Broadly speaking, all recommender systems use some kind of knowledge to predict whether a certain item is suitable for a certain target. Following Burke [6], we can mention four main prediction techniques: *content-based* (CB), *collaborative filtering* (CF), *social*, *knowledge-based* (KB), and *hybrid recommender systems*.

Content-based recommendation can be seen as a machine learning *classification* task, where the recommender identifies potentially interesting items based on their similarity to items the target users "liked" in the past [39]. In *collaborative filtering* approaches [44], evaluation predictions are computed based either on the ratings provided by users with a similar behaviour with respect to the target, or on the ratings of items that are characterized by a similar pattern of evaluations. *Social* or *community-based* recommender systems take into account the ratings of friends instead of those of unknown users [47].

In this paper we focus on *knowledge-based* recommender systems which explicitly represent requirements on candidate items. Knowledge bases are used to encode information on items and users, as well as on relationships between item and user features (domain model). Relationships can refer, for example, to compatibility constraints among user interests and items, and among items themselves. Knowledge-based recommendation significantly emphasizes the role of the domain model. Various approaches have been used to represent this type of knowledge, ranging from rules (mapping items to be recommended to user features and preferences) to stereotypes or overlay models (associating properties of users to items to be recommended), ontologies (describing the domain and possibly associated with stereotypes or overlay models), dependency (causal) networks, ... If compared with the other approaches, knowledge-based recommender systems certainly imply higher costs for knowledge acquisition [4]. On the other hand, however, they do not require to handle historical data (such as ratings or buying behaviour) neither about the target nor about other users.

Hybrid recommender systems combine features from different recommendation techniques to overcome their respective limitations and improve overall

performance. For example, a knowledge-based approach could complement a collaborative filtering algorithm [6] in order to mitigate the so-called "cold start" problem, namely the fact that the system is unable to generate recommendations for new users and new items.

2.2. Families of recommender systems

In the following subsections, we will briefly outline several families of recommender systems, which can use any of the aforementioned techniques.

2.2.1. Context-aware recommenders

Context-aware recommenders [1] provide recommendations that depend on both user preferences and contextual information. Context has been variously defined in the literature. Schilit et al. [45] defined context as location, nearby people and things, and changes which happen to them. Brown et al. [5] considered location, nearby people, time, season, and temperature. Dey [18] defined context as "any information that can be used to characterize the situation of an entity". Adomavicius et al. [1] identified temporal context, physical context, social context, interaction media context (the device in use), and modal context (which represents the current state of mind of the user, the user's goals, mood, experience, and cognitive capabilities). Here, we define as *context* all the external conditions surrounding the target users.

2.2.2. Contraint-based recommenders

Constraint-based recommenders [21] are a special class of knowledge-based ones. They use constraints as a powerful formalism to represent compatibilities/incompatibilities and/or relations among user features, context features and recommendation items. Users explicitly specify their preferences in terms of item properties and these are internally represented as rules.

2.2.3. Negative-preferences based recommendations

While most recommenders rely on information on what users like, some systems also take into account information on what users do not like or, even more strongly, dislike. Negative preferences can be used to warn users against items to avoid [50] or, similarly to constraints, to filter candidate recommendations [29].

2.2.4. Package recommendations

With *package* recommendations, each suggestion consists in a set of items which are expected to be consumed "together" [3]. Examples of packages are playlists of songs or movies, travel plans or sets of points of interest which can be visited as part of a single trip, bundles of products which can be bought together, as well as teams of players or co-workers or a single item, such as a restaurant, accompanied by a group of people with whom that item can be enjoyed [30]. *Sequence* recommendations (see, e.g., [42]) extend the package concept by including temporal constraints on item consumption.

2.2.5. Group recommendations

Starting from the assumption that several items, such as restaurants or movies, are likely to be used by groups as often as by individuals, group recommenders face the problem of *aggregating* information about single users to adapt to a group as a whole. Different strategies can be adopted (see [33] for a review): for example, systems which prioritize the *maximization of average satisfaction* can calculate some sort of average of the individual predicted ratings/preferences; systems which aim at *minimizing misery*, i.e., avoiding that any member is very dissatisfied, can recommend items only if the lowest individual rating is higher than a certain threshold; systems which aim at *maximizing pleasure* can select items based on their highest individual rating or on the mere number of preferences they accumulated (*plurality voting*). More advanced approaches try to take into account social aspects such as disagreement/consensus among group members, differences in user personalities (e.g., cooperative vs.

assertive) and social connections (see, e.g., [43]).

3. Characterization of recommendation problems

This section introduces a logical characterization of a *knowledge-based recommendation problem*.

The knowledge base of a recommender consists of three main ingredients:

- A vocabulary of features concerning the user;
- A vocabulary of features concerning the context;
- A vocabulary of features concerning the items to be recommended.

Moreover, it includes relations among these features[1], which will be represented as logical formulae according to the pattern:

features of the item to be recommended and of the context
$\rightarrow$ features of the user

specifying that an item to be recommended is suitable, in a given context, for some features concerning user preferences. This is, in fact, a convenient way to express the aforementioned relations in logic.

For example, the fact that a pizza restaurant is a suitable recommendation for people who are interested in Italian food for dinner can be represented through the formula:

restaurant(pizza) AND time(dinner) $\rightarrow$ interest(italian_food)

Definition 1. *A* **recommendation problem** *RP is a pair*

$$RP = \langle DM, OBS \rangle,$$

[1]Notice that, in this paper, we abstract from the source of such a knowledge base, which could either be provided by an expert or be learnt from examples. In fact, the here presented formalization is independent from this aspect.

where:

- DM *is a set of logical formulae, involving the following set of predicates:*
  - F *is a set of predicates representing user features and preferences,*
  - R *is a set of predicates representing items among which the one(s) to be recommended have to be chosen,*
  - C *is a set of predicates representing contextual features,*
- $OBS = \langle UF, CXT \rangle$ *is a set of ground atoms denoting the specific case to be solved and*
  - UF *represents features and preferences of a specific user*
  - CXT *represents a specific contextual situation*

As a running example, we take inspiration from a recommender system we designed in the past, iCity [8], which suggests activities to carry out in the city of Turin. For the sake of simplicity, we will focus only on restaurant recommendations.

$$
\begin{aligned}
F = \{\ & dietary\_requirements(X) - where\ X\ can\ assume\ the\ values: \\
& \qquad vegetarian, vegan, no\_dairy, coeliac, ..., \\
& \qquad no\_dietary\_requirements \\
& food\_preferences(X) - where\ X\ can\ assume\ the\ values: \\
& \qquad traditional, ethnic, fast\_food, ... \\
& spending\_style(X) - where\ X\ can\ assume\ the\ values: \\
& \qquad low, medium, high \\
& personal\_style(X) - where\ X\ can\ assume\ the\ values: \\
& \qquad formal, informal \\
& food\_aversions(X) - where\ X\ can\ be\ any\ type\ of\ food\ or\ food\ category \\
& ambient\_aversion(X) - where\ X\ can\ assume\ the\ values: \\
& \qquad extreme\_lighting, extreme\_noises, extreme\_smells, ..., \\
& \qquad no\_ambient\_aversions \\
\}
\end{aligned}
$$

$$R = \{\ restaurant\_A,\ restaurant\_B,\ restaurant\_C\ \}$$

$$\begin{aligned}
C = \{\ & occasion(X) - where\ X\ can\ assume\ the\ values: \\
& \qquad business, romantic, party, ... \\
& company(X) - where\ X\ can\ assume\ the\ values: \\
& \qquad alone, small_group, large_group \\
& meal(X) - where\ X\ can\ assume\ the\ values: \\
& \qquad lunch, dinner, breakfast \\
& time_availability(X) - where\ X\ can\ assume\ the\ values: \\
& \qquad in_a_hurry, no_time_constraints \\
\}
\end{aligned}$$

$$\begin{aligned}
DM = \{\ & food(Japanese) \rightarrow food_preference(ethnic) \\
& style(informal) \rightarrow personal_style(informal) \\
& food(pizza) \rightarrow food_preference(traditional) \\
& price(high) \rightarrow spending_style(high_spending), \\
& style(romantic)\ AND\ occasion(romantic) \rightarrow ambient_aversion(no_aversion), \\
& food(fast_food)\ AND\ time_availability(no_time_constraints) \\
& \rightarrow personal_style(informal), \\
& style(formal)\ AND\ meal(dinner) \rightarrow personal_style(formal), \\
& light(extreme_lighting)\ AND\ meal(lunch) \rightarrow personal_style(informal), \\
& restaurant_A \rightarrow food(Japanese)\ AND\ price(high)\ AND\ light(candlelit) \\
& \ AND\ noise(silent), \\
& restaurant_B \rightarrow food(pizza)\ AND\ style(informal), \\
& restaurant_C \rightarrow food(fastfood)\ AND\ price(low)\ AND\ smells(extreme)\ \}
\end{aligned}$$

Solving a recommendation problem consists in finding a set of recommendation items (atoms in R) which are in accordance with the set of atoms in OBS, or, more specifically, which are in accordance with user preferences UF, given the context CXT. Hence, we can provide a first abstract characterization of the notion of *solution* to a recommendation problem.

Definition 2. *Given a recommendation problem* $RP = \langle DM, OBS \rangle$, $OBS = \langle UF, CXT \rangle$. *A set* S *of ground instances of predicates in* R *is a solution to* RP *if and only if* $DM \cup S \cup CXT$ *is in accordance with* UF.

Informally, we are stating that a set of items can be suggested to a user as a recommendation if they are somehow related to their preferences, in the given contextual situation.

Let us consider the running example above. Given the observations OBS:

- $UF = \{$
  $user\_1 = \{$ $dietary\_requirements(vegetarian)$, $food\_preferences(ethnic)$, $spending\_style(high\_spending)$, $personal\_style(informal)$, $food\_aversions(onion)$, $ambient\_aversions(no\_ambient\_aversions)$, $...\}$
  $\}$

- $CXT = \{$
  $CXT\_X = \{occasion(romantic)$, $meal(dinner), ...\}$
  $\}$

According to definition 2, a solution which is in accordance with the preferences of $user_1$ in the model is $S = \{restaurant\_A\}$.

4. A spectrum of logical definitions of recommendation

Definition 2 is informal about the notion of *being in accordance with*: in this section we will discuss how it can be formalized. Bearing in mind the fact that we are interested in recommending those items which, in the logical models, are related to observed user preferences - or, in other words, account for them - there are different ways to translate this notion into logic. At one extreme, it could be formalised as *logical entailment*, requiring that the recommended items entail the user preferences, in the given context. At the other extreme it could be formalised as *logical consistency*, requiring that the items do not predict anything which is inconsistent with user preferences.

This alternative reminds the same debate which arose in the community of model-based reasoning when the notion of diagnosis and, in particular, that of *explaining observations* was formalised, leading to the consistency-based and abductive theories of diagnosis [27]. Finding the diagnoses that explain the observations may in fact be formalized as either choosing those faults that entail (predict) the observations (strong notion of explanation as causation, i.e. abduction in logic) or choosing those faults that do not entail (predict) something which is in contrast to what is observed (weak notion of explanation as consistency).

In this section we suggest that a similar definition and similar discussions can be applied also to the recommendation task, leading to a spectrum of definitions of the notion of solution to the recommendation problem.

In particular, the idea is that the two notions of explanation can be put together in a definition which covers both of them. More precisely, the set of observations UF can be partitioned into two subsets UF^A and UF^C and we assume that different notions of *being in accordance with* can be adopted for the two subsets: a strong one (entailment) for the former and a weak one (consistency) for the latter. By varying the way UF is partitioned we can move from a definition which requires entailment for all user preferences (when all preferences are in UF^A and UF^C is empty) to the the other extreme - weak notion for all user preferences - in the opposite case. This leads to the following parametric definition:

Definition 3. *Given a recommendation problem* $RP = \langle DM, OBS \rangle$*, where* $OBS = \langle UF, CXT \rangle$*. Let us consider a partitioning of* UF *in two subsets* UF^A *and* UF^C*. A set* S *of ground instances of predicates in* R *is a solution to* RP *if and only if*

- $DM \cup S \cup CXT \models UF^A$
- $DM \cup S \cup CXT$ *is consistent with* UF^C*.*

The definition is parametric with respect to the partitioning of UF:

- At one extreme there is the case where UF^A=UF and $UF^C = \emptyset$. This corresponds to a strong notion of recommendation where *in accordance with* means *entails* (Abductive recommendation).
- At the other extreme there is the case where UF^C=UF and $UF^A = \emptyset$. This corresponds to a weak notion of recommendation where *in accordance with* means *consistent with* (consistency-based recommendation).
- In between there is a lattice of alternatives where $UF^A \subset UF$. In particular the lattice is defined by the subset partial order on the set UF^A of the user preferences that have to be entailed. Thus the abductive approach is the top of the lattice and the consistency-based is the bottom.

Interestingly, the following property can be proved.

Property 1. *Given a recommendation problem RP and let RP_1 and RP_2 be two reformulations of RP such that UF_1^A is the subset of UF to be entailed in RP_1 and UF_2^A is the subset of UF to be entailed in RP_2 and let Sol_1 and Sol_2 be the sets of solutions to RP_1 and RP_2, then we have that:*

$$\textit{if } UF_1^A \subseteq UF_2^A \textit{ then } Sol_2 \subseteq Sol_1$$

In other words, whenever some user data D are moved from UF^C to UF^A, the set of solutions shrinks and those that do not entail D are removed (but no extra solutions are added). This proves that interpreting *in accordance with* as entailment leads to a stronger and more restrictive notion of recommendation and that the spectrum is a lattice with a weak definition at the bottom and a strong one at the top and a complete range of alternatives in between. It is worth noting that definition 3 corresponds to the one adopted for unifying the formalizations of model-based diagnosis; thus, all the results and techniques developed in that field also apply to the the case of recommendation. In the following we transpose the most significant ones to our case.

4.1. Choosing the appropriate definition

Given a recommendation problem, one might wonder whether there are criteria for choosing the most appropriate definition or it is simply a matter of preference of the designer. The literature of model-based diagnosis showed that the choice between abductive and consistency-based diagnosis depends mainly on properties of the model of the system to be diagnosed [14]. Similar considerations can also be applied to the case of recommendation problems.

A first relevant property is the completeness of the model. By completeness here we mean that the model DM contains a complete set of relations among R (the recommendations), C (contextual data) and F (the user features). In other words, DM is a faithful representation of reality and no information (especially no relation) which is relevant in the reality is missing.

Thesis 1. *If the model DM is complete, then abduction is the best choice and consistency produces spurious solutions. Conversely, if the model is not complete, abduction is too restrictive and consistency-based should be preferred.*

The underlying reason can be explained intuitively. If the set R contains all the items r to be recommended which are related to a user preference a and the DM contains an implication $r \rightarrow a$ for each one of such items r, then no other item s is a suitable recommendation for a even if s is consistent with a in the model. Thus s is a spurious solution. On the other hand, if we cannot assume that the model is complete and, for example, a relation between a recommendation s and a may be missing, then limiting to the relations in the model may be too restrictive and consistency provides plausible solutions.

In other words, the more complete is the model, the stronger is the notion of recommendation that should be used; the less complete is the model, the weaker is the notion of recommendation that we can afford.

Let us present a simple example that explains the statement above. Consider the aforementioned domain model, as well as a specific user, whom we will call $user\_2$:

$$
\begin{aligned}
UF = \{ & \\
& user\_2 = \{dietary\_requirements(no\_dietary\_requirements), \\
& food\_prefereces(pizza),\ spending\_style(low\_spending), \\
& personal\_style(informal),\ food\_aversions(fish), \\
& ambient\_aversions(no\_ambient\_aversions),\ ...\} \\
\}
\end{aligned}
$$

Since $user\_2$ likes pizza ($food\_preference(pizza)$), then abduction produces only one solution $S_1 = \{restaurant\_B\}$, which corresponds to the only pizza restaurant in R. In contrast, consistency-based produces also the extra solution $S_2 = \{restaurant\_C\}$, a fast food, which is not in contrast with any of $user\_2$ features.

In the assumption that the model is complete, the solution S_2 is spurious and should not be provided to the user. On the other hand, if the model is not complete, then abduction is too restrictive and other solutions may be missing. For example, if a relation between $food(fast\_food)$ and $food\_preference(pizza)$ is missing, then abduction erroneously excludes the solution S_2 from the solution sets while consistency-based includes it.

The completeness of the model may be partial in the sense that some parts of the model may be complete while others may be more uncertain and may be missing some relevant relations. In this case one should choose an intermediate definition, which is abductive with respect to the preferences that are in the complete part of the model.

Similar considerations apply to the completeness of data concerning a specific problem to be solved (user preferences, contextual data).

From a different point of view, abduction corresponds to making a closed world assumption on the model [13]. This means assuming that everything which is not mentioned is false (or, alternatively, is not of interest for the application being developed). In some cases this assumption is preferable to circumscribe the set of solutions avoiding to produce a wide open set of alternatives

which in a sense are less useful as they may involve or consider aspects that have not been modelled explicitly in the knowledge base.

4.2. Weak (conditional) solutions and uncertain models

An important remark concerns uncertainty in the model. Although in this paper we do not consider quantitative approaches to uncertainty (such as probabilities or fuzzy logic), logical models and our spectrum of definitions allow us to deal with some qualitative forms of uncertainty in the model.

Firstly, the spectrum of definitions allows a designer to cope with models which are incomplete or at least partially incomplete by choosing the appropriate notion of recommendation (i.e. moving towards a consistency-based approach) which takes into account if and which parts of the model (and of user data) may be incomplete.

This can be even done incrementally. One may start from an abductive definition and then progressively relax if the set of proposed solutions is too limited. Relaxation can be performed by moving user preferences from the set of those to be implied to the set of those for which only consistency is required. Heuristics can suggest which preferences should be moved.

A further approach can be again borrowed from model-based diagnosis: Console et al. [13] show that the formulae in the model can be extended by adding literals which represent a qualitative form of uncertainty to their premise.

Whenever the formula

$r \rightarrow a$

representing the relation between an item to be recommended r and a user preference a is uncertain, this can be represented by changing the formula above into

$r \rightarrow \wedge \alpha a$

where α is an atom representing uncertainty, i.e., the fact that the relation is not a strong implication. This can be regarded as a qualitative form of uncertainty, i.e., a qualitative probability that the item r is suitable for the user preference a.

The predicates representing uncertainty correspond to assumptions that may be made in order to generate solutions. In other words this means that in order to suggest r as a recommendation whenever the user has the preference a, the assumption α has to be made, that is, a solution is:

$S = \{r\}$ under the set of assumptions $As = \{\alpha\}$

Thus, in a sense, S is a weak (or conditional) solution or, in other words, a solution subject to an assumption or, from a different point of view, a qualitative probability. The larger is the set of assumptions, the weaker is a solution and, thus, these assumptions represent qualitative uncertainty of the solution and provide a way of ranking solutions in a qualitative way.

This leads to the following extension of the definition of recommendation.

Definition 4. *Given a recommendation problem $RP = \langle DM, OBS\rangle$, where $OBS = \langle UF, CXT\rangle$, where UF is partitioned in the subsets UF^A and UF^C.*

Let A be a set of literals denoting uncertain information which can appear in the premises of the formulae in DM.

A set S of ground instances of predicates in R in conjunction with the assumption of a set As of literals in A is a solution to RP if and only if

- $DM \cup S \cup CXT \cup As \models UF^A$
- $DM \cup S \cup CXT \cup As$ *is consistent with* UF^C.

Thus As is the set of assumptions of uncertain knowledge that have to be made in order to be in accordance with the observation. The smaller this set, the more "certain" a solution is.

For example, let us assume that in our model the relation between the restaurant feature $food(fast\_food)$ and the user's preference $personal\_style(informal)$ is uncertain. This can be represented by replacing the formula

$food(fast\_food) \rightarrow personal\_style(informal)$

with the formula:

$food(fast\_food) \wedge \alpha \rightarrow personal\_style(informal)$

where α is an atom representing uncertainty.

Thus, adopting a strong definition of recommendation and given the following observation:

$$
\begin{aligned}
UF = \{ & \\
& user\_2 = \{dietary\_requirements(no\_dietary\_requirements), \\
& food\_prefereces(pizza),\ spending\_style(low\_spending), \\
& personal\_style(informal),\ food\_aversions(fish), \\
& ambient\_aversions(no\_ambient\_aversions),\ ...\} \\
\}
\end{aligned}
$$

we can compute two solutions:

$S_1 = \{restaurant\_B\}$

$S_2 = \{restaurant\_C\}$ under the set of assumptions $As = \{\alpha\}$

This makes S_1 preferable since it does not involve making any assumption on uncertain knowledge.

4.3. Complexity and efficient algorithms

The above discussed formalizations have been used for providing complexity analyses of the diagnostic problem solving task, which is not surprisingly inherently intractable. More interestingly, a huge number of approaches and algorithms for dealing with or controlling this complexity have been proposed.

The same algorithms could be adopted for designing knowledge-based recommenders. For example, some approaches exploit a truth maintenance system [15], others exploit fault probabilities to guide the process [16] (in the case of recommendation, this approach would exploit probability estimates for the items to be recommended); others are based on the pre-compilation of guiding heuristics [12], and still others exploit OBDD [49]. All of them show that diagnoses can be effectively computed and indeed the community established a competition for the design of efficient diagnosers (see [19]).

5. Recommendation as a process

The literature on diagnosis provides methodologies for characterising the task as a process where candidate solutions are progressively refined and ranked. The methodologies can be adapted to the recommendation task.

5.1. Discriminating among alternative solutions

A recommendation problem RP has typically more than one solution. Given the set Sol of solutions, many criteria can be adopted to discriminate among them. The approach proposed by deKleer in [16] can be adapted: given a set of solutions, it can suggest which, among the yet unknown user preferences, should be elicited to best discriminate among the solutions, either directly asking users by means of a dialogue, as in conversational recommender systems [34], or learning from user behaviour with the systems, or using preferences of similar users. This approach can also exploit information about the cost/benefit of each solution (e.g., in terms of risks and/or of repair action) or the cost of eliciting data and information on *a priori* distribution of faults, if available (but these are not necessary).

5.2. Ranking solutions

The recommendation process discussed in the previous subsection may close without producing a single solution but rather a set of solutions that cannot be further discriminated. At this point some criteria to rank the solutions should be used in order to select the one (or ones) to be presented to the user. Also in this case the literature in the area of model-based diagnosis can provide interesting criteria and insights.

Structural criteria. A first group of criteria is based on structural properties of the solutions which generate a partial ordering among the set Sol of solutions (we introduce the symbol $\prec$ and the notation $S_1 \prec S_2$ to represent that S_1 precedes S_2 in the ordering).

- *Subset minimality.* A first criterion is to rank solutions according to the relation among the sets of items involved in each solution. In detail, given two solutions S_1 and S_2, we have that $S_1 \prec S_2 \; iff \; S_1 \subset S_2$.

- *Minimal cardinality.* A second criterion is to rank solutions according to their cardinality, i.e., the number of items they involve. In detail, given two solutions S_1 and S_2, we have that $S_1 \prec S_2 \; iff \; \parallel S_1 \parallel < \parallel S_2 \parallel$

It is worth making a consideration at this point. In model-based diagnosis the items in each solution represent faults and thus the solutions that involve a minimal number (or a minimal cardinality) of faults should be preferred as the simultaneous occurrence of multiple faults is usually not very common.

In the case of recommender systems the situation is not as clear and the preference for minimality can be questioned or it can be domain and application dependent. In fact, one may argue that a maximal (or maximal cardinality) solution should be preferred as it involves a more specific recommendation with respect to a minimal one (see also Section 6.5 on package recommendation).

For example, given the two following solutions:

$$S_1 = \{restaurant\_A, restaurant\_B\}$$
$$S_2 = \{restaurant\_A, restaurant\_B, restaurant\_C\}$$

according to the minimal cardinality criterion, $S_1 \prec S_2$. S_2 might be the best option since it has a better recall. On the other hand, one may also argue that minimal solutions are less constraining on users, giving them a perception of more degrees of freedom and thus should be preferred. In the example, S_2 can be perceived as too challenging for users asking for some suggestions to spend the night, so that they can prefer a lighter and more viable recommendation like S_1.

Our characterization is neutral with respect to the above criteria and the designers may choose the most appropriate one for their domain and/or application.

Domain/application dependent criteria. Other more specific criteria, peculiar to a domain or application or based on probabilistic/statistic considerations can be adopted. For example, one may prefer solutions which involve specific items or one may define partial ordering among items based on domain specific considerations such as costs or other metric information (probability) associated with the items (e.g., how often they have been appreciated in the past by the user or by similar users or by the community of users in general).

Qualitative uncertainty. Finally, the approach discussed in section 4.2 provides another way of ranking solutions, based on the set of assumptions that have to be made in each one of the solutions. At a finer grain level also the assumption literals could be ranked to express different levels of qualitative probabilities and this can provide an even finer way of ranking the solutions, leading to choosing those which involve a minimum amount of qualitative uncertainty.

6. A variety of approaches to recommendation

In this section, we analyze how our definition can naturally cope with some variations of the recommendation problem from the literature (see Section 2).

6.1. "Basic" recommendation

Many recommenders exploit knowledge bases to compute single user - single item recommendations (see Section 2). Depending on the inference strategy being adopted, each approach can be mapped to one of the definitions in the spectrum. For example, categorical inference with rules corresponds in most cases to the strong definition of recommendation based on entailment, while in the case of stereotypes and overlay models the definition being adopted depends on the form of match between prototypes and user data, ranging from weaker definitions (in case partial match is accepted) to stronger ones (in case a full match is required). Thus, our framework provides a theoretical basis for comparing these systems.

6.2. Context-aware recommendation

Context awareness (see Section 2.2.1) is included in definition 3 where each solution must be in accordance with CXT.

6.3. Negative preferences based recommendation

The use of negative preferences (see Section 2.2.3) is covered quite naturally by our general definition. Since in this case we aim at recommendations which do not predict items that correspond to what the user does not like, this can be easily dealt as follows: if user data include a negative preference p, then $\neg p$ must be added to UF^C.

We can reformulate the definition of solution:

Definition 5. *Given a recommendation problem $RP = \langle DM, OBS \rangle$, where $OBS = \langle UF, CXT \rangle$, where $UF = \langle POS, NEG \rangle$ distinguishes between positive and negative user preferences. Let us consider a partitioning of POS in two subsets UF^A and UF^C and let NEG^- be the set of the negations of the negative preferences in NEG. A set S of ground instances of predicates in R is a solution to RP if and only if*

- $DM \cup S \cup CXT \models UF^A$
- *$DM \cup S \cup CXT$ is consistent with $UF^C \cup NEG^-$.*

The second condition in the definition requires that the proposed solution is consistent with the negation of the negative observation, i.e. that it does not entail the negative observations. The rest is unchanged: thus, the definition above is parametric and we have a spectrum of alternatives for recommendation with negative preferences.

Let us take into account user 3, who loves informal restaurants, but cannot stand places with extreme smells:

$$
\begin{aligned}
UF = \{ & \\
& user\_3 = \{dietary\_requirements(no\_dietary\_requirements), \\
& food\_preferences(traditional),\ spending\_style(high\_spending),
\end{aligned}
$$

$personal\_style(informal)$, $food\_aversions(anchovies)$,

$ambient\_aversions(extreme\_smells)$, ...}

}

Thus, we have:

$NEG = \{ambient\_feature(extreme\_smells)\}$

and thus:

$NEG^{-} = \{\neg ambient\_feature(extreme\_smells)\}$

If we did not consider negative preferences, and took into account the fact that the target user likes informal restaurants, we would have two solutions $S_1 = \{restaurant\_B\}$ and $S_2 = \{restaurant\_C\}$. However, considering negative preferences, only S_1 is a solution according to definition 5.

6.4. Constraint-based recommendation

Constraints on user data (see Section 2.2.2) can be simply dealt with in the same way as negative observations in the sense that a solution must satisfy the further condition of being consistent with all the available constraints, i.e., it must not make predictions which violate each one of the constraints. For example, there can be constraints related to context and user preferences: a user that likes alcoholic drinks may not appreciate to have them in the morning. This constraint can be formally expressed as follows:

$\neg(food\_preferences(alcoholics) \wedge time(morning))$.

More generally, since constraints can be expressed in logical terms we can extend the definition of the domain model to include the formulae representing this type of knowledge. Thus all the definitions above can naturally deal with this type of knowledge without any change.

6.5. Package and sequence recommendation

In our approach, given a recommendation problem RP, a package (see Section 2.2.4) is generated with no need for further adjustments whenever there is a

solution $S \in Sol$ and $\| S \| > 1$. Consistency and other constraints between the items in each package are dealt with as any other constraints (see Section 6.4). Typical examples of constraints can refer to package cardinality or to the total "cost" of a certain solution (in a trip planning recommender, see e.g. [3], each item can be associated to a cost in terms of price or time to visit, and the total cost of a package can be understood as a budget or trip duration constraint). Sequence recommendations can be generated in case recommended items have temporal labels and temporal constraints are available (notice that our definition could be extended to this case, opening another correspondence with the formalisation of temporal diagnosis and thus further interesting insights for the recommendation task, see the section on dynamic systems in [27]).

Notice that, if the set R of candidate items can be partitioned into different subsets $R_1, R_2, \ldots R_n$ that identify different types of items (for example, "restaurants", "hotels", "events"), we may formulate a set of constraints on package composition, requiring that a valid package is a set $\{r_1, r_2, \ldots r_n\}$ where $r_1 \in R_1, r_2 \in R_2, \ldots r_n \in R_n$, and no two items belong to the same subset.

6.6. Group recommendation

The formalization of group recommendations (see Section 2.2.5) varies according to the chosen aggregation strategy. In the following, we will analyze a couple of common approaches in order to show how they can be covered by our definition.

- *One solution fits for all users.* A variant of *plurality voting,* this approach aims at finding a solution which is in accordance with the preferences of all users, provided that these preferences are not in contrast. A set of group features and preferences GF can be built as the union of the features and preferences UFs of individual users. The recommendation should then be based on GF, replacing UF by GF in definition 3. An equivalent way to determine the set of solutions is to determine the solutions for each user independently and then merge them, for example intersecting them.

In the context of our leading example, we will consider a group that consists in users 2 and 3, with the following observations:

- For user 2: $UF_2^A = \{dietary\_requirements(no\_dietary\_requirements)$, $food\_preferences(pizza)$, $spending\_style(low\_spending)$, $personal\_style(informal)$ $\}$;
  $NEG_2^- = \{\neg food\_preferences(fish)\}$
- For user 3: $UF_3^A = \{dietary\_requirements(no\_dietary\_requirements)$, $food\_preferences(traditional)$, $spending\_style(high\_spending)$, $personal\_style(informal)$ $\}$;
  $NEG_3^- = \{\neg food\_preferences(anchovies), \neg ambient\_preferences(extreme\_smells)\}$

Possible solutions for user 3 are $Sols_3 = \{\{restaurant\_A\}, \{restaurant\_B\}, \{restaurant\_C\}\}$, while possible solutions for user 2 are $Sols_2 = \{\{restaurant\_B\}, \{restaurant\_C\}\}$: therefore, a set of solutions for the whole group can be determined as $Sols_{2,3} = Sols_2 \cap Sols_3 = \{\{restaurant\_B\}, \{restaurant\_C\}\}$.

- *Least misery.* This strategy can be formalized by adopting the approach to compute recommendations with negative preferences using as negative preferences the union of what all users dislike.

  In this case the solution to group recommendation can be defined as the intersection of individual recommendations with negative preferences.
  As an example, let us now assume that a group consists in users 1 and 2, with the following observations:

  - For user 1: $UF_1^A = \{dietary\_requirements(vegetarian)$, $food\_preferences(ethnic)$, $spending\_style(high\_spending)$, $personal\_style(informal)$ $\}$;
    $NEG_1^- = \{\neg food\_preferences(onion)\}$
  - For user 2: $UF_2^A = \{dietary\_requirements(no\_dietary\_requirements)$, $food\_preferences(pizza)$, $spending\_style(low\_spending)$, $personal\_style(informal)$ $\}$;
    $NEG_2^- = \{\neg food\_preferences(fish)\}$

Possible solutions for user 1 are $Sols_1 = \{\{restaurant\_A\}, \{restaurant\_B\}\}$, while possible solutions for user 2 are $Sols_2 = \{\{restaurant\_B\}, \{restaurant\_C\}\}$: therefore, a set of solutions for the whole group is $Sols_{1,2} = \{\{restaurant\_B\}\}$, since other solutions implying a preference for costly ($restaurant\_A$) or non-vegetarian ($restaurant\_C$) restaurants would be unacceptable for either group member.

These approaches can be weakened in many ways. The set of users for which the solution is computed can be reduced, looking for maximal (or maximal cardinality or sets that are maximal according to some priority criteria) sets of users whose preferences are consistent and for whom solutions can be computed. Any of the definitions of solution to the recommendation problem for individuals can be adopted.

In those cases where a solution that satisfies all the users in the group cannot be found, it may be interesting to suggest also which is the "best group" for each one of the identified solutions, thus generating "mixed" packages of groups and items, similarly to [30].

7. Related work

While related work can be identified primarily in other problem solving tasks where logic has been used for formalization purposes, in the following we will also discuss other ways in which logic has been used so far in recommender systems.

Logic-based problem solving formalizations. As pointed out in the introduction, many problem solving tasks benefited from formal characterizations based on some form of logic. This area of research dates back to the early days of artificial intelligence, when tasks such as game playing and planning were characterized in terms of logic (a very interesting and seminal discussion in favour of logical foundations of problem solving can be found in [26]). Starting from the 80s' the case of diagnosis has been paradigmatic and has opened several areas of

investigation, as we have already extensively discussed. The parallel between diagnosis and recommendation we established in this paper allows us to extend all the results which were obtained thanks to these characterizations to the case of knowledge-based recommendation.

Similar transpositions were made for other problem solving tasks. For example, logical theories of configuration [20] were proposed relying on the consistency-based approach to diagnosis. This led to significant results, including the design of efficient commercial systems. Analogous considerations apply to system design [25], where definitions similar to those adopted for diagnosis characterize solutions which are in accordance with models describing components to be assembled and their relations and constraints. This correspondence also opened new interesting trends of research and application in the areas of sensor placing (during design) and design for diagnosability (i.e., the ability to design systems that will be easily diagnosable and repairable/ re-configurable during operations).

The logical definition of diagnosis was also used to characterize a variety of problems in software analysis, from early work on debugging logic programs [11], or hardware design via hdl [24], to more recent work on diagnosis of software requirements and diagnosis and self repair of web services [23].

From a different perspective, the adoption of formal frameworks has proved to be beneficial in many areas of artificial intelligence. In particular, abduction provided a basis for a number of activities and gave rise to a number of problem solving frameworks. Paradigmatic is the case of abductive logic programming, which extends logic programming with abductive reasoning (see [17] for an introduction and overview). It has been applied to a variety of problem solving tasks, including ontology management, scheduling, business process management, learning and database analysis.

These considerations motivated us to analyse the recommendation task from a logical perspective, trying to reformulate it in the same terms of the above-mentioned tasks. This led to the results discussed in the previous sections which represent, in our view, a starting point rather than a target. The short

discussion in this section, in fact, shows that many other interesting issues and results can be transposed to recommendation from other logical formalizations of problem solving. We will return to this in the conclusion.

Characterizations of recommendation. Few other approaches have been proposed in the literature to characterize recommender systems; they, however, do not cover the variety of dimensions we considered. Felfernig et al. in [21] proposed a characterization of recommender systems as constraint satisfaction problems. Knowledge is represented as a set of constraints involving items to be recommended, user preferences and features, contextual features. Given a set of user and contextual data, computing a solution amounts to finding the recommendation items for which the constraints are satisfied. This approach can be easily mapped to our framework. In fact, constraints can be represented as logical formulae, while constraint consistency and satisfaction can be mapped to our logical definitions of solution.

Logic-based recommendations. Logic was also exploited by some recommender systems to develop specific internal reasoning components, especially based on fuzzy logic, which is useful to describe uncertainty in rating prediction [32, 37, 38, 7, 10]. However, these approaches differ from ours, in that we use logic as a means to provide a conceptualisation for the recommendation problem and to support the analysis and comparison of different systems, not to develop reasoning modules.

8. Conclusions and discussions

This paper introduced a logical formalization for knowledge-based recommender systems, singling out a spectrum of definitions of solution to a recommendation problem. Such spectrum covers various approaches to recommendation and provides a principled way for comparing them, based on the semantics of the recommendation process itself. It is important to notice that we do not

aim at introducing yet another approach for solving the recommendation problem, but rather at providing a unifying view, a sort of meta level definition which covers a wide range of existing approaches and which allows to analyse some properties of the recommendation task at knowledge level. Since the here adopted formalisation has been borrowed from the one we proposed for diagnostic problem solving, a significant result of this work is that all the literature concerning the formal approaches to diagnosis can be mapped to recommendation, providing both theoretical insights which extend those in this paper in various directions and efficient approaches and strategies for implementing problem solvers in accordance with the definition.

A major limitation of the formalization we proposed is that, being based on logic, it does not cope with quantitative uncertainty. In the paper we discussed how to deal with a qualitative form of uncertainty in the knowledge base, associating a qualitative conditional probability to the relation between items to be recommended and user preferences. This approach could be easily extended to deal with multiple levels of qualitative uncertainty.

Quantitative extensions of the diagnostic framework adopted in this paper have been proposed in the literature, starting from the seminal work of Pearl on causation [40] and of Peng and Reggia on causal probabilistic abduction [41]; many approaches for probabilistic model based diagnosis have been proposed based on Bayesian networks, see for example the work by Lucas on abductive diagnosis [31] or the one by Darwiche et al. [35].

In the future we plan to extend our characterization to deal with some form of quantitative uncertainty; in particular, we will investigate at least two options for extending our domain models and consequently the characterization of solution: on the one hand, we will follow the trends above and represent the relations in the model using Bayesian networks; on the other hand, we will also consider the option of moving to fuzzy logic, following the approaches discussed in the related work section. These extensions, however, are outside the scope of this paper.

Another problem which is open for future research is the one concerning the

validation of a knowledge base. Currently most of the approaches are based on user testing where the results generated by the system are compared to those provided by a sample of users. The logical formalization of recommendation could be the basis for a more theoretic approach to validation, exploiting methodologies from model checking. This issue, however, is also outside the scope of this paper.

Interestingly, the ability to describe and compare recommender systems from a semantic point of view is also very relevant if we adopt a user-centric perspective. In fact, being able to characterize recommendations based on their meaning makes them inherently more transparent and easier to explain, an issue which has been gaining increasing relevance [48]. In addition, different definitions of recommendation answer different user needs, and are therefore appropriate for specific domains and applications. For example, in Section 6 we have shown how variations of the recommendation problem can be dealt with by choosing appropriate definitions in the spectrum, e.g., for dealing with what the user dislikes (recommendation with negative preferences) or for dealing with package or group recommendation. From a broader perspective, let us consider the issue of computer credibility and trust building. According to Fogg [22], one of the contexts where computer credibility is relevant for interaction is when computers give advice or provide instructions to their users, as is the case for recommender systems. Credibility perceptions usually evolve over time: in particular, Kantowitz et al. [28] empirically confirmed the idea that computers strengthen their credibility when they provide correct information, while they lose their credibility otherwise. Fogg [22] found that small errors can have disproportionately large effects on perceived credibility. From this perspective, the choice of a definition of solution (see Section 4.1) can be reframed as the choice of the type of error a recommender system is less unwilling to make - ideally, the one which is less likely to affect its perceived credibility: either missing a potentially useful solution (in the case of an abductive definition, if the model is not complete), or providing a factually incorrect suggestion (in the case of a consistency-based definition, if the model is complete).

In conclusion, we can claim that not only does the formalization we proposed provide a means to assess recommenders based on their use of the available knowledge and the semantics of the recommendation process itself, but it also creates a bridge with problem solving tasks, which can be useful to foster research and application in the area of knowledge-based recommendation.

References

[1] G. Adomavicius, R. Sankaranarayanan, S. Sen, A. Tuzhilin, Incorporating contextual information in recommender systems using a multidimensional approach, ACM Trans. Inf. Syst. 23 (1) (2005) 103–145. doi:10.1145/1055709.1055714.
URL http://doi.acm.org/10.1145/1055709.1055714

[2] M. Beladev, L. Rokach, B. Shapira, Recommender systems for product bundling, Knowledge-Based Systems 111 (2016) 193 – 206. doi:https://doi.org/10.1016/j.knosys.2016.08.013.
URL http://www.sciencedirect.com/science/article/pii/S0950705116302751

[3] I. Benouaret, D. Lenne, A package recommendation framework for trip planning activities, in: Proceedings of the 10th ACM Conference on Recommender Systems, RecSys '16, ACM, New York, NY, USA, 2016, pp. 203–206. doi:10.1145/2959100.2959183.
URL http://doi.acm.org/10.1145/2959100.2959183

[4] S. Bouraga, I. Jureta, S. Faulkner, C. Herssens, Knowledge-based recommendation systems:a survey, International Journal of Intelligent Information Technologies 10 (2014) 1–19. doi:10.4018/ijiit.2014040101.

[5] P. J. Brown, J. D. Bovey, X. Chen, Context-aware applications: from the laboratory to the marketplace, IEEE Personal Communications 4 (5) (1997) 58–64. doi:10.1109/98.626984.

[6] R. D. Burke, Hybrid web recommender systems, in: The Adaptive Web, Methods and Strategies of Web Personalization, 2007, pp. 377–408.

[7] Y. Cao, Y. Li, An intelligent fuzzy-based recommendation system for consumer electronic products, Expert Systems with Applications 33 (1) (2007) 230–240.

[8] F. Carmagnola, F. Cena, L. Console, O. Cortassa, C. Gena, A. Goy, I. Torre, F. Vernero, Tag-based user modeling for social multi-device adaptive guides, User Model. User-Adapt. Interact. 18 (5) (2008) 497–538.

[9] Ò. Celma, The Long Tail in Recommender Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 87–107. doi:10.1007/978-3-642-13287-2_4.
URL https://doi.org/10.1007/978-3-642-13287-2_4

[10] F. Cena, L. Console, C. Gena, A. Goy, G. Levi, S. Modeo, I. Torre, Integrating heterogeneous adaptation techniques to build a flexible and usable mobile tourist guide, AI Commun. 19 (4) (2006) 369–384.
URL http://content.iospress.com/articles/ai-communications/aic386

[11] L. Console, G. Friedrich, D. T. Dupré, Model-based diagnosis meets error diagnosis in logic programs, in: R. Bajcsy (Ed.), Proceedings of the 13th International Joint Conference on Artificial Intelligence. Chambéry, France, August 28 - September 3, 1993, Morgan Kaufmann, 1993, pp. 1494–1501.

[12] L. Console, L. Portinale, D. T. Dupré, Using compiled knowledge to guide and focus abductive diagnosis, IEEE Trans. Knowl. Data Eng. 8 (5) (1996) 690–706.

[13] L. Console, D. Theseider Dupré, P. Torasso, On the relationship between abduction and deduction, J. Log. Comput. 1 (5) (1991) 661–690. doi:10.1093/logcom/1.5.661.
URL https://doi.org/10.1093/logcom/1.5.661

[14] L. Console, P. Torasso, A spectrum of logical definitions of model-based diagnosis, Computational Intelligence 7 (1991) 133–141. doi:10.1111/j.1467-8640.1991.tb00388.x.
URL https://doi.org/10.1111/j.1467-8640.1991.tb00388.x

[15] J. de Kleer, Problem solving with the ATMS, Artif. Intell. 28 (2) (1986) 197–224. doi:10.1016/0004-3702(86)90082-2.
URL https://doi.org/10.1016/0004-3702(86)90082-2

[16] J. de Kleer, Using crude probability estimates to guide diagnosis, Artif. Intell. 45 (3) (1990) 381–391. doi:10.1016/0004-3702(90)90012-0.
URL https://doi.org/10.1016/0004-3702(90)90012-0

[17] M. Denecker, A. C. Kakas, Special issue: abductive logic programming, J. Log. Program. 44 (1-3) (2000) 1–4. doi:10.1016/S0743-1066(99)00078-3.
URL https://doi.org/10.1016/S0743-1066(99)00078-3

[18] A. K. Dey, Understanding and using context, Personal Ubiquitous Comput. 5 (1) (2001) 4–7. doi:10.1007/s007790170019.
URL http://dx.doi.org/10.1007/s007790170019

[19] A. Feldman, J. de Kleer, T. Kurtoglu, S. Narasimhan, S. Poll, D. García, L. D. Kuhn, A. J. C. van Gemund, The diagnostic competitions, AI Magazine 35 (2) (2014) 49–54.

[20] A. Felfernig, G. Friedrich, D. Jannach, M. Stumptner, Consistency-based diagnosis of configuration knowledge bases, Artif. Intell. 152 (2) (2004) 213–234. doi:10.1016/S0004-3702(03)00117-6.
URL https://doi.org/10.1016/S0004-3702(03)00117-6

[21] A. Felfernig, G. Friedrich, D. Jannach, M. Zanker, Constraint-based recommender systems, in: Recommender Systems Handbook, 2015, pp. 161–190. doi:10.1007/978-1-4899-7637-6_5.
URL https://doi.org/10.1007/978-1-4899-7637-6_5

[22] B. J. Fogg, H. Tseng, The elements of computer credibility, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '99, ACM, New York, NY, USA, 1999, pp. 80–87. doi:10.1145/302979.303001.
URL http://doi.acm.org/10.1145/302979.303001

[23] G. Friedrich, M. Fugini, E. Mussi, B. Pernici, G. Tagni, Exception handling for repair in service-based processes, IEEE Trans. Software Eng. 36 (2) (2010) 198–215. doi:10.1109/TSE.2010.8.
URL https://doi.org/10.1109/TSE.2010.8

[24] G. Friedrich, M. Stumptner, F. Wotawa, Model-based diagnosis of hardware designs, Artif. Intell. 111 (1-2) (1999) 3–39. doi:10.1016/S0004-3702(99)00034-X.
URL https://doi.org/10.1016/S0004-3702(99)00034-X

[25] M. P. J. Fromherz, D. G. Bobrow, J. de Kleer, Model-based computing for design and control of reconfigurable systems, AI Magazine 24 (4) (2004) 120–130.
URL http://www.aaai.org/ojs/index.php/aimagazine/article/view/1735

[26] M. R. Genesereth, N. J. Nilsson, Logical foundations of artificial intelligence, Morgan Kaufmann, 1988.

[27] W. Hamscher, L. Console, J. de Kleer (Eds.), Readings in Model-based Diagnosis, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1992.

[28] B. H. Kantowitz, R. J. Hanowski, S. C. Kantowitz, Driver acceptance of unreliable traffic information in familiar and unfamiliar settings, Human Factors 39 (2) (1997) 164–176. doi:10.1518/001872097778543831.
URL https://doi.org/10.1518/001872097778543831

[29] D. H. Lee, P. Brusilovsky, Reinforcing recommendation using implicit negative feedback, in: G.-J. Houben, G. McCalla, F. Pianesi, M. Zancanaro (Eds.), User Modeling, Adaptation, and Personalization, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 422–427.

[30] I. Lombardi, F. Vernero, What and who with: A social approach to double-sided recommendation, Int. J. Hum.-Comput. Stud. 101 (2017) 62–75. doi:10.1016/j.ijhcs.2017.01.001.
URL https://doi.org/10.1016/j.ijhcs.2017.01.001

[31] P. J. F. Lucas, Bayesian model-based diagnosis, Int. J. Approx. Reasoning 27 (2) (2001) 99–119.

[32] Q. Madera, O. Castillo, M. García-Valdez, A. Mancilla, A method based on interactive evolutionary computation and fuzzy logic for increasing the effectiveness of advertising campaigns, Information Sciences 414 (2017) 175–186.

[33] J. Masthoff, Selecting news to suit a group of criteria: An exploration, in: Proceedings of the 4th Personalized TV workshop, associated with AH 2004: International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Springer, 2004.

[34] L. McGinty, B. Smyth, On the role of diversity in conversational recommender systems, in: International Conference on Case-Based Reasoning, Springer, 2003, pp. 276–290.

[35] O. J. Mengshoel, M. Chavira, K. Cascio, S. Poll, A. Darwiche, N. S. Uckun, Probabilistic model-based diagnosis: An electrical power system case study, IEEE Trans. Syst. Man Cybern. Part A 40 (5) (2010) 874–885. doi:10.1109/TSMCA.2010.2052037.
URL https://doi.org/10.1109/TSMCA.2010.2052037

[36] A. Newell, The knowledge level, Artif. Intell. 18 (1) (1982) 87–127. doi:

10.1016/0004-3702(82)90012-1.
URL https://doi.org/10.1016/0004-3702(82)90012-1

[37] B. Ojokoh, M. Omisore, O. Samuel, T. Ogunniyi, A fuzzy logic based personalized recommender system, International Journal of Computer Science and Information Technology & Security (IJCSITS) 2 (2012) 1008–1015.

[38] H. Pandey, V. Singh, A fuzzy logic based recommender system for e-learning system with multi-agent framework, International Journal of Computer Applications 122 (17).

[39] M. Pazzani, D. Billsus, Learning and revising user profiles: The identification ofinteresting web sites, Machine Learning Journal 27 (3) (1997) 313–331. doi:10.1023/A:1007369909943.

[40] J. Pearl, Heuristics - intelligent search strategies for computer problem solving, Addison-Wesley series in artificial intelligence, Addison-Wesley, 1984.

[41] Y. Peng, J. A. Reggia, A probabilistic causal model for diagnostic problem solving part I: integrating symbolic causal inference with numeric probabilistic inference, IEEE Trans. Syst. Man Cybern. 17 (2) (1987) 146–162. doi:10.1109/TSMC.1987.4309027.
URL https://doi.org/10.1109/TSMC.1987.4309027

[42] M. Quadrana, P. Cremonesi, D. Jannach, Sequence-aware recommender systems, ACM Comput. Surv. 51 (4) (2018) 66:1–66:36. doi:10.1145/3190616.
URL http://doi.acm.org/10.1145/3190616

[43] L. Quijano-Sanchez, J. A. Recio-Garcia, B. Diaz-Agudo, G. Jimenez-Diaz, Social factors in group recommender systems, ACM Trans. Intell. Syst. Technol. 4 (1) (2013) 8:1–8:30. doi:10.1145/2414425.2414433.
URL http://doi.acm.org/10.1145/2414425.2414433

[44] J. B. Schafer, D. Frankowski, J. L. Herlocker, S. Sen, Collaborative filtering recommender systems, in: The Adaptive Web, Methods and Strategies of Web Personalization, 2007, pp. 291–324.

[45] B. N. Schilit, M. M. Theimer, Disseminating active map information to mobile hosts, IEEE Network 8 (5) (1994) 22–32. doi:10.1109/65.313011.

[46] T. Silveira, M. Zhang, X. Lin, Y. Liu, S. Ma, How good your recommender system is? a survey on evaluations in recommendation, International Journal of Machine Learning and Cybernetics 10 (2019) 813–831.

[47] R. R. Sinha, K. Swearingen, Comparing recommendations made by online systems and friends, in: DELOS, 2001.

[48] N. Tintarev, J. Masthoff, Designing and Evaluating Explanations for Recommender Systems, Springer US, Boston, MA, 2011, pp. 479–510. doi:10.1007/978-0-387-85820-3_15.
URL https://doi.org/10.1007/978-0-387-85820-3_15

[49] G. Torta, P. Torasso, On the role of modeling causal independence for system model compilation with obdds, AI Commun. 20 (1) (2007) 17–26.

[50] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, S. Ma, Explicit factor models for explainable recommendation based on phrase-level sentiment analysis, in: Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR '14, ACM, New York, NY, USA, 2014, pp. 83–92. doi:10.1145/2600428.2609579.
URL http://doi.acm.org/10.1145/2600428.2609579